廣東工業大學

QG 中期考核详细报告书

题　　目　中期考核报告书

学　　院　自动化学院

专　　业　数据科学与大数据技术

年级班别　20 级 1 班

学　　号　3220001512

学生姓名　罗宇彤

年　月　日

目录

准备工作

1. 导入库（库如下）

```
import pandas as pd
import missingno as msno
import matplotlib.pyplot as plt
from sklearn.feature_selection import SelectKBest, f_classif
import numpy as np
from sklearn.ensemble import AdaBoostClassifier
from sklearn.linear_model import Perceptron
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

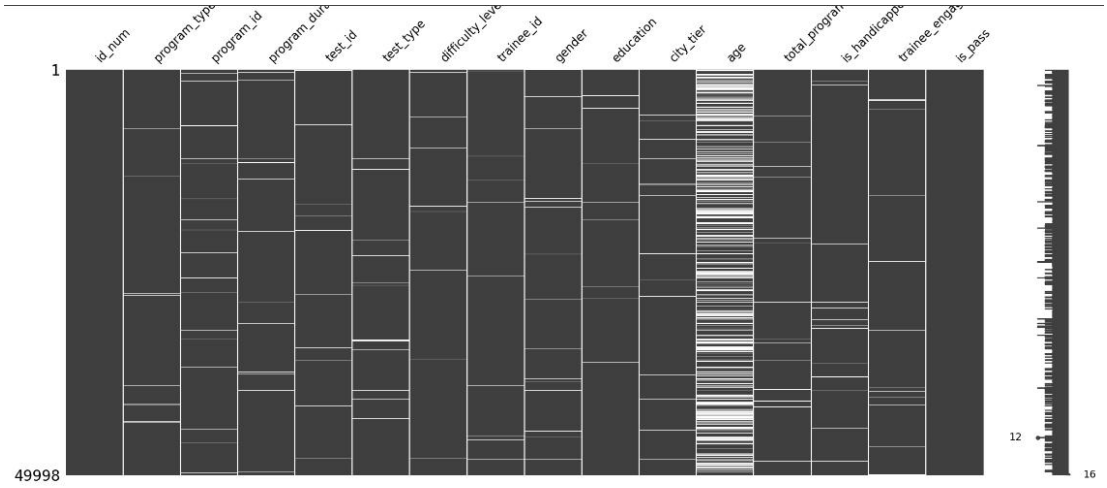
2. 读取数据：通过 pd.read_csv()读取数据

数据清洗

1. 查看数据信息：用 dataframe.info()查看总体信息，如下图：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49998 entries, 0 to 49997
Data columns (total 16 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   id_num                      49998 non-null  object
 1   program_type                49998 non-null  object
 2   program_id                  49299 non-null  object
 3   program_duration            49998 non-null  float64
 4   test_id                     49273 non-null  float64
 5   test_type                   49998 non-null  object
 6   difficulty_level            49998 non-null  object
 7   trainee_id                  49259 non-null  float64
 8   gender                      49998 non-null  object
 9   education                   49998 non-null  object
 10  city_tier                   49998 non-null  float64
 11  age                         49998 non-null  float64
 12  total_programs_enrolled     49998 non-null  float64
 13  is_handicapped              49998 non-null  object
 14  trainee_engagement_rating   49998 non-null  float64
 15  is_pass                     49998 non-null  int64
dtypes: float64(7), int64(1), object(8)
```

2. 检查缺失值：通过 dataframe.isnull().sum().sort_values(ascending=False)的方法检查缺失值的情况，同时还使用了 missingno 库中的 matrix()函数。前者能够反馈具体数值，而后者则以图的形式反映出来，更直观。如下图：



3. 填补数据：先使用 pandas 中的 unique()函数去重，查看各特征下包含的值，若数据不全，则将进行填补。对于缺失情况不是特别严重的特征，就采取用该特征下出现次数最多的数据进行填补，而对于缺失情况比较严重的特征（如 age），无论使用均值填充或出现次数最大的数填充，都会有较大的误差，故考虑删除该特征，不进行填补工作。
4. 再次检查数据的缺失情况

特征选择

1. 对数据进行独热编码：在使用 unique()查看去重后的数据后，就知道了该列所包含的内容有什么。有些不一定是数字，则要对其进行独热编码。例如在 gender 特征中以 0 表示 M,以 1 表示 F。在后面的特征中，含有字符的，也进行相应的处理。

2. 所有数据中，一共有 15 个特征，维数大，并且不是所有特征都有用，故对其进行筛选。查看数据之后，发现，用户的 id 号码基本上是随机的，对训练模型来说，是无用特征，故先将有关 id 的特征删除掉，再对剩下的特征进行进一步筛选。

3. 留出法划分训练集和测试集：为了更好地训练模型，将原本的 train 数据集又进行划分，分别用作训练和测试。

4. 归一化处理：由于不同特征值下的数据可能存在单位或者尺度不同的问题，因此在训练模型的过程中，所占尺度更大的特征可能会占更大的权重，尤其在后面选择模型的时候，采用的是 adaboost 进行分类，为了消除各特征之间单位和尺度的差异，故对剩下的特征进行了归一化处理。

5. 计算特征得分，筛选数据：使用了 selectkbest 筛选出得分较高的特征，在此基础上再去除得分低的特征，最后把剩下的数据作为训练的数据。

模型选择

使用了 adaboost 算法，其中是以感知机为弱分类器。采取这种方法是希望通过集成学习的方法建立起一个强分类器，在这个过程中提高那些被前一轮弱分类器错误分类样本的权值，而降低那些被正确分类样本的权值。这样，误分类点会因为权值变大了而受到更大的关注，从而更准确地分类。

1. 从 sklearn.linear_model 导入 Perceptron，并创建一个感知机
2. 创建 adaboost 分类器，并将感知机以弱分类器传入，同时设置弱分类器的个数及学习

率

3. 开始训练：使用 adaboost 下的 fit()函数，传入训练集
4. 用测试集检验：使用 adaboost 下的 predict()函数，传入前面分好的测试集，进行测试。

模型评估

使用 classification_report()对训练结果进行评估。其中参数为前面分好的测试集的真实值和预测结果。

最后，（某一次的运行数据）得到准确率为 0.68。其中在这份具有 15000 组数据的数据集中，预测出来为 0 的有 4596 份，但是精确率却只有 0.45，召回率只有 0.14，而 1 的召回率和精确率都相对较高，说明预测出来的结果中，预测为 1 的占比比较大。

这个结果在后面传入新的数据集进行预测的时候，同样得到验证：预测出来的结果往往是偏向一种结果。

对于感知机的个人理解

由数据可知，每一个(x,y)都有一个对应的标签，即其所对应的真实值。以 x+y>1 则对应 1，反之对应 0 为例。

因此，感知机就是对传入的数据进行分类，判断它对应的是 1 还是 0。因此我的目的就是要找到一个“超平面”，这个平面可以把所有数据划分为两部分，一部分对应的是 1，另一部分对应的是 0.

既然是求“平面”，则要先求出它的法向量，设其为 weights，同时，该平面在空间中不一定过原点，故还要加上偏置 bias，因此，训练的目的就是得到 weights 和 bias。

在获取较为准确的 weights 和 bias 的过程，我理解为是一个不断拟合的过程。首先，先给 weights 和 bias 赋初值，再将点带入当前的“平面”方程中，看这个点是在平面的上面还是下面。如果原本应该在下面的点代进方程之后，反馈说它在上面，则说明判断错了，即 weights 和 bias 的值不对，要进行更新。

而判断这个点是在当前平面的上面还是下面，就要通过一个激活函数来判断。此处，由于前一轮的数据集都是非负的数，所以我选取了符号函数，即返回值为+1 或-1，能更好地帮助我理解感知机。对于在平面上方的点，返回+1，反之返回-1。假设(0.5,0.5)原本在平面上方，计算到它就在平面上方，得到返回值为 1，那么此时在 weights 和 bias>=0 的情况下，(weights*(0.5,0.5)+bias)*1>0，若计算到它在平面下方，即返回-1，此时，(weights*(0.5,0.5)+bias)*(-1)<0，小于 0 就说明错了，要调整 weights 和 bias。
要使这个平面比较拟合，即要所有点到这个平面的距离最小化，也就是通过调整 weights 和 bias 的值来使总距离最小。这里得到的总距离公式可作为该模型的损失函数，要使距离最小，即要求该函数的最小值。此处用到随机梯度下降法，因为避免数值太多导致遍历太慢。随机选取一个误分类点，使其梯度下降。求出梯度之后更新 weights 和 bias。另外作补充的是，在得出预测值之后，将其与真实值进行比较，把数值不同的取出来，即误分类点单独成列，每一次迭代中都是从误分类点中随机抽取一个点。经过足够多次迭代后，weights 和 bias 就会收敛（对于可分开为两类的数据来说）。最后就可以进行预测了。

关于考核任务的个人想法

按理说，使用 adaboost 算法会比单独使用单层感知机的效果要好，但是在这项任务中，我得到的结果却表现为严重偏向某一方，当然其中不排除原本的数据集就存在比例分配不合理的因素。

询问了其他人，之后，认为可能是迭代的次数不够多。因为在这项任务中，感知机是以导入库的方式创建的，就不像自己写的感知机那样，存在许多漏洞。因此，可能是因为迭代次数不够多，而原本的学习率相比于适合的值是在一个较大的值上，此时采取梯度下降的方法容易让模型把预测结果都变成同一个类别，若增加迭代次数，可能会使其回到正常值。

除此之外，在前面特征工程的步骤中，可能选择出来的特征还不具备代表性。因为在选择之前，进行的缺失值的处理，都是通过填补出现次数最多的值，这样可能会出现较大的误差，猜想使用回归的方法，对每一个缺失的值进行预测之后再填补可能效果会更好，更有助于特征的选择。

资料

为了对 adaboost 和集成学习有更深的了解，泛读了一篇论文，论文信息如下：

文献名：Emsemble Methods in Machine Learning

作者：Thomas G.Dietterich

会议：1st International Workshop on Multiple Classifier Systems (MCS 2000)

他引次数：7180

该篇论文首先指出了许多学习算法存在的不足之处，如

1. Statistical

与假设空间的大小相比，当用于训练的数据量太小时，就会出现统计问题。

没有足够的数据，学习算法能够在空间 H 中找到许多不同的分类器进行预测分类，而它们的预测出来的准确率是相同的。通过集成的方法，建立一个集成的分类器，此时利用学习算法能够平均各分类器的结果同时降低了我们在进行预测时恰好选择了一个错误的分类器的概率。

2. Computational

许多学习算法会采取局部搜索，结果可能会导致进入局部最优。此时，可以尝试从多个点出发，进行搜索，可能会取得比较准确的结果。

3. Representational

大多数时候，函数 f 不能被空间中的分类器拟合到。

由此进一步说明采用集成学习的方法有望减少甚至消除这些缺陷。

后面又介绍了构建集成学习的方法，如 Bagging, Adaboost 等。

最后对不同的集成方法进行比较，得到结论：在低噪音的情况下，AdaBoost 可以提供出色的性能，因为它能够优化缩小合奏而不会过度拟合。但是，在高噪声情况下，AdaBoost 将大量的关注点放在标签错误的示例上，这导致过度拟合非常严重。

Bagging 和随机处理在嘈杂和无噪声的情况下处理得都挺好，因为它们专注于统计问题和噪声增加了这个统计问题。在大量数据中，随机化可能会比 Bagging 做得更好，因为大量的 Bootstrap 复制品训练集与训练集本身非常相似。